To – do app

```
import { useState } from "react"; // Importing useState hook from React
import {
  View,        // Importing View component for layout
  Text,       // Importing Text component for displaying text
  TextInput,    // Importing TextInput component for user input
  TouchableOpacity, // Importing TouchableOpacity for button-like behavior
  FlatList,    // Importing FlatList for rendering lists
  StyleSheet,   // Importing StyleSheet for styling components
} from "react-native"; // Importing necessary components from React Native

// Main App component
const App = () => {
  // State to store the current task input
  const [task, setTask] = useState("");
  // State to store the list of tasks
  const [tasks, setTasks] = useState([]);
  // State to track the index of the task being edited
  const [editIndex, setEditIndex] = useState(-1);

  // Function to handle adding or updating a task
  const handleAddTask = () => {
    if (task) {
      if (editIndex !== -1) {
        // If editIndex is not -1, update the existing task
        const updatedTasks = [...tasks];
        updatedTasks[editIndex] = task; // Update the task at the specified index
        setTasks(updatedTasks); // Update the tasks state
        setEditIndex(-1); // Reset editIndex
      } else {
        // If editIndex is -1, add a new task
        setTasks([...tasks, task]); // Add the new task to the tasks array
      }
      setTask(""); // Clear the input field
    }
  };
```

```
// Function to handle editing a task
const handleEditTask = (index) => {
  const taskToEdit = tasks[index]; // Get the task to be edited
  setTask(taskToEdit); // Set the task in the input field
  setEditIndex(index); // Set the index of the task being edited
};

// Function to handle deleting a task
const handleDeleteTask = (index) => {
  const updatedTasks = [...tasks];
  updatedTasks.splice(index, 1); // Remove the task at the specified index
  setTasks(updatedTasks); // Update the tasks state
};

// Function to render each task item
const renderItem = ({ item, index }) => (
  <View style={styles.task}>
    <Text style={styles.itemList}>{item + " "}</Text> {/* Display the task */}
    <View style={styles.taskButtons}>
      <TouchableOpacity onPress={() => handleEditTask(index)}>
        <Text style={styles.editButton}>Edit</Text> {/* Edit button */}
      </TouchableOpacity>
      <TouchableOpacity onPress={() => handleDeleteTask(index)}>
        <Text style={styles.deleteButton}>Delete</Text> {/* Delete button */}
      </TouchableOpacity>
    </View>
  </View>
);

// Main UI rendering
return (
  <View style={styles.container}>

    <Text style={styles.title}>ToDo App</Text> {/* App title */}
    <TextInput
```

```
        style={styles.input}
        placeholder="Enter task" // Placeholder text for the input field
        value={task} // Bind input value to task state
        onChangeText={(text) => setTask(text)} // Update task state on text change
      />

      <TouchableOpacity
        style={styles.addButton}
        onPress={handleAddTask} // Call handleAddTask on button press
      >
        <Text style={styles.addButtonText}>
          {editIndex !== -1 ? "Update Task" : "Add Task"} {/* Button text changes based on
edit mode */}
        </Text>
      </TouchableOpacity>

      <FlatList
        data={tasks} // Pass tasks array as data
        renderItem={renderItem} // Render each task using renderItem
        keyExtractor={(_item, index) => index.toString()} // Unique key for each item
      />
    </View>
  );
};

// Styles for the components
const styles = StyleSheet.create({
  container: {
    flex: 1, // Take full screen height
    padding: 40, // Add padding around the container
    marginTop: 40, // Add margin at the top
  },
  title: {
    fontSize: 24, // Font size for the title
    fontWeight: "bold", // Bold font
    marginBottom: 20, // Margin below the title
```

```
},

input: {
  borderWidth: 3, // Border width for the input field
  borderColor: "#ccc", // Border color
  padding: 10, // Padding inside the input field
  marginBottom: 10, // Margin below the input field
  borderRadius: 10, // Rounded corners
  fontSize: 18, // Font size for the input text
},
addButton: {
  backgroundColor: "green", // Background color for the button
  padding: 10, // Padding inside the button
  borderRadius: 5, // Rounded corners
  marginBottom: 10, // Margin below the button
},
addButtonText: {
  color: "white", // Text color
  fontWeight: "bold", // Bold font
  textAlign: "center", // Center align the text
  fontSize: 18, // Font size for the button text
},
task: {
  flexDirection: "row", // Arrange items in a row
  justifyContent: "space-between", // Space between items
  alignItems: "center", // Align items vertically in the center
  marginBottom: 15, // Margin below each task
  fontSize: 18, // Font size for the task text
},
itemList: {
  fontSize: 19, // Font size for the task text
},
taskButtons: {
  flexDirection: "row", // Arrange buttons in a row
},
editButton: {
```

```
    marginRight: 10, // Margin to the right of the edit button
    color: "green", // Green color for the edit button text
    fontWeight: "bold", // Bold font
    fontSize: 18, // Font size for the edit button text
  },
  deleteButton: {
    color: "red", // Red color for the delete button text
    fontWeight: "bold", // Bold font
    fontSize: 18, // Font size for the delete button text
  },
});

// Export the App component as the default export
export default App;
```